UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www uspto gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/087,027	02/28/2002	Adam W. Smith	MS1-0861USC1	6939

22801 7590 02/17/2009
LEE & HAYES, PLLC
601 W. RIVERSIDE AVENUE
SUITE 1400
SPOKANE, WA 99201

EXAMINER	
ANYA, CHARLES E	

ART UNIT	PAPER NUMBER
2194	

MAIL DATE	DELIVERY MODE
02/17/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 10/087,027	**Applicant(s)** SMITH ET AL.	
	Examiner CHARLES E. ANYA	**Art Unit** 2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *10 November 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-41* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-41* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *11/11/08*.	6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-41 are pending in this application.

Claim Objections

2. **Claims 19-23 are objected to because of the following informalities:**

It appears claim 19 invokes a 112 6th. paragraph however, it is not clear from the specification as to what the "means for" represents. It is advisable for Applicant to clearly indicate/claim what the "mean for" represents (i.e. whether is a software/hardware). The current amendment seems to suggest that the "means for" is software, for the invocation of 112 6th. paragraph to be directed to a statutory subject matter the "means for" has to be a hardware (i.e. processor).

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. **Claims 1-8, 10-16, 19-22, 24-29, 31-34 and 36-39 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pub. No. 2006/0294500 A1 to Chiang in view of U.S. Pat. No. 6,792,605 B1 to Roberts et al.**

5. As to claim 1, Chiang teaches a software architecture implemented at least in part by a computing device for a distributed computing system comprising:

a plurality of applications configured to handle requests submitted by remote devices over a network, wherein the plurality of applications are written in different programming languages (Web Application 400 page 3 paragraphs 0033, "…input files…" page 3 paragraphs 0035/0036, Input 605 page 4 paragraph 0039);

an application program interface to present functions used by the plurality of applications to access network and computing resources of the distributed computing system (Application Framework 410 page 3 paragraph 0033); and

a common language runtime layer that translates the plurality of applications written in different programming languages into an intermediate language (Web Application Source Code Output 610), the intermediate language being: executed natively by the common language runtime layer ("The web application generator 205 generates the application framework code 605…regardless of the language format for the input files, because the input tags are interpreted in the same fashion for the different languages…") and configured to access resources or services requested by the remote devices whereby a seamless and robust integration between multi-language application development is allowed (Web Application Generator 205 page 4 paragraphs 0039 – 0044, page 5 paragraph 0050, page 6 paragraphs 0064, page 7 paragraphs 0068/0069).

Chiang is silent with reference to providing secure execution environment for multiple programming languages.

Roberts teaches providing secure execution environment for multiple programming languages ("...access control..." Col. 4 Ln. 36 – 38, Col. 6 Ln. 1 – 9, Ln. 47 – 63).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the system of Chiang with the teaching of Roberts because the teaching of Roberts would improve the system of Chiang by providing the essential services of *identification and authentication* (*I&A*), *authorization*, and *accountability* where identification and authentication determine who can log on to a system, and the association of users with the software subjects that they are able to control as a result of logging in; authorization determines what a subject can do and accountability identifies what a subject (or all subjects associated with a user) did.

6. As to claim 2, Chiang teaches the software architecture as recited in claim 1, wherein the distributed computing system comprises client devices and server devices that handle requests from the client devices, the remote devices comprising at least one client device (figures 1/2).

7. As to claim 3, Chiang teaches the software architecture as recited in claim 1, wherein the distributed computing system comprises client devices and server devices that handle requests from the client devices, the remote devices comprising at least one server device that is configured as a Web server (figures 1/2).

8. As to claim 4, Roberts teaches the software architecture as recited in claim 1, wherein the application program interface comprises: a first group of services related to creating Web applications (Col. 7 Ln. 50 – 67, Col. 9 Ln. 27 – 35);

a second group of services related to constructing client applications (Col. 14 Ln. 30 – 46); a third group of services related to data and handling XML documents (Col. 10 Ln. 1 – 9, Ln. 59 – 67); and

a fourth group of services related to base class libraries (Col. 6 Ln. 7 – 9, Col. 8 Ln. 29 – 38, Ln. 64 – 67).

9. As to claim 5, see the rejection of claims 1 and 4 above.

10. As to claim 6, Roberts teaches the application program interface as recited in claim 5, wherein the first group of services comprises:

first functions that enable construction and use of Web services (Col. 9 Ln. 27 – 35);

second functions that enable temporary caching of frequently used resources (Col. 11 Ln. 1 – 5);

third functions that enable initial configuration (Col. 7 Ln. 11 – 15);

fourth functions that enable creation of controls and Web pages (Col. 14 Ln. 30 – 46);

fifth functions that enable security in Web server applications (Col. 6 Ln. 7 – 9, Ln. 48 – 67, Col. 7 Ln. 50 – 56);

sixth functions that enable access to session state values (Col. 6 Ln. 23 – 27).

11. As to claim 7, Roberts teaches the application program interface as recited in claim 5, wherein the second group of services comprises:

first functions that enable creation of windowing graphical user interface; and

second functions that enable graphical functionality (Col. 14 Ln. 30 – 46).

12. As to claim 8, Roberts teaches the application program interface as recited in claim 5, wherein the third group of services comprises:

first functions that enable management of data from multiple data source (Col. 5 Ln. 25 – 43); and

second functions that enable XML processing (Col. 5 Ln. 25 – 37, Col. 10 Ln. 1 – 9, Ln. 59 – 67).

13. As to claims 10 and 11, see the rejection of claims 5 and 1 respectively.

14. As to claim 12, Roberts teaches the distributed computer software architecture as recited in claim 11, further comprising a remote application configured to be executed on one of the remote computing devices, the remote application using the application programming interface to access the networking platform (figure 1 Web Service Engine 101 Col. 4 Ln. 60 – 67, Col. 5 Ln. 1 – 25).

15. As to claims 13-16, see the rejection of claims 4, 6 and 7.

16. As to claim 19, Roberts teaches the system comprising:

one or more microprocessors; and

a memory storing one or more software programs comprising computer-executable instructions executable by the one or more microprocessors (although the Roberts prior art does not explicitly indicate one or more microprocessors and a memory it is well known in that one or more microprocessors and a memory are inherent in providing web based services, as is the case with the Robert prior art), the one or more software programs comprising:

means for exposing a set of functions that enable browser/server communication; means for exposing a second set of functions that enable drawing and construction of client applications (Col. 14 Ln. 30 – 46);

means for exposing a third set of functions that enable connectivity to data sources and XML functionality (Col. 5 Ln. 25 – 37, Col. 10 Ln. 1 – 9, Ln. 59 – 67);

means for exposing a fourth set of functions that enable system and runtime functionality (Col. 8 Ln. 22 – 28) and providing secure execution environment for multiple programming languages is provided ("…access control…" Col. 4 Ln. 36 – 38, Col. 6 Ln. 1 – 9, Ln. 47 – 63).

Chiang teaches means for translating Web applications written in different languages into an intermediate language, the intermediate language being: executed natively by the common runtime layer ("The web application generator 205 generates the application framework code 605...regardless of the language format for the input files, because the input tags are interpreted in the same fashion for the different languages...") and configured to access resources or services, whereby a seamless and robust integration between multi-language application development is allowed (Web Application Generator 205 page 4 paragraphs 0039 – 0044, page 5 paragraph 0050, page 6 paragraphs 0064, page 7 paragraphs 0068/0069).

17. As to claims 20-22, see the rejection of claims 6-8 respectively.

18. As to claims 24, 31 and 36, see the rejection of claim 19 above.

19. As to claim 25, Roberts teaches the computer implemented method as recited in claim 24, further comprising receiving a request from a remote computing device, the request containing a call to at least one of the first, second, third, and fourth functions (Col. 5 Ln. 1 – 25).

20. As to claim 26, see the rejection of claims 20-23 above.

21. As to claims 27-29, see the rejection of claims 6-8 respectively.

22. As to claims 32-34, see the rejection of claims 6-8 above.

23. As to claims 37-39, see the rejection of claim 6-8 respectively.

24. Claims 9, 17, 18, 23, 30, 35 and 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pub. No. 2006/0294500 A1 to Chiang in view of U.S. Pat. No. 6,792,605 B1 to Roberts et al. as applied to claim 5, and further in view of U.S. Pat. No. 5,987,517 to Firth et al.

25. As to claim 9, Roberts teaches the application program interface as recited in claim 5, wherein the fourth group of services comprises:

first functions that enable definitions of various collections of objects (Col. 8 Ln. 50 – 67);

fifth functions that enable input/output of data (Col. 8 Ln. 29 – 38, Ln. 64 – 67);

sixth functions that enable a programming interface to network protocol (figure 1 Col. 4 Ln. 60 – 67, Col. 5 Ln. 25 –37);

eleventh functions that enable character encoding (inherent in XML language, since XML language supports character encoding);

ninth functions that enable system security and permissions (Col. 6 Ln. 7 – 9);

tenth functions that enable installation and running of services (Col. 9 Ln. 27 – 35); and

thirteenth functions that facilitate runtime operations (Col. 8 Ln. 22 – 28).

Roberts and Chiang are silent with reference to second functions that enable programmatic access to configuration settings and handling of errors in configuration files; third functions that enable application debugging and code execution tracing; fourth functions that enable customization of data according to cultural related information; seventh functions that enable a managed view of types, methods, and fields; eighth functions that enable culture-specific resources and twelfth functions that enable multi-threaded programming;

Firth teaches second functions that enable programmatic access to configuration settings and handling of errors in configuration files/third functions that enable application debugging and code execution tracing (Col. 13 Ln. 6 – 29); fourth functions that enable customization of data according to cultural related information/eighth functions that enable culture-specific resources (Col. 13 Ln. 6 – 29); seventh functions that enable a managed view of types, methods, and fields; twelfth functions that enable multi-threaded programming (Col. 12 Ln. 52 – 62);

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify of system of Roberts and Chiang with the teaching of Firth because the teaching of Firth would improve the system of Roberts and Chiang by creating computer network applications by using a library of reentrant network functions which allow an application to reduce the source code required to interact with a computer network such as the internet (Firth Col. 1 Ln. 9 – 14).

26. As to claims 17, 23, 30, 35 and 40 see the rejection of claim 9 above.

27. As to claims 18, Chiang teaches a computer system including one or more microprocessors and one or more software programs, that are written in different languages and utilize an application program interface to request services from an operating system through a common language runtime layer (Application Framework 410 page 3 paragraph 0033) and the common language runtime layer that allows multi-language development, with cross language inheritance and translates the one or more software programs written in different programming languages into an intermediate language, wherein the intermediate language is executed natively by the common runtime layer ("The web application generator 205 generates the application framework code 605…regardless of the language format for the input files, because the input tags are interpreted in the same fashion for the different languages…") and is configured to access to the services requested by the one or more software programs (Web Application Generator 205 page 4 paragraphs 0039 – 0044, page 5 paragraph 0050, page 6 paragraphs 0064, page 7 paragraphs 0068/0069).

Roberts teaches the application program interface including separate commands to request services consisting of the following groups of services:

A. a first group of services related to creating Web applications, the first group of services comprising:

constructing Web services (Col. 9 Ln. 27 – 35);

temporary caching resources (Col. 11 Ln. 1 – 5);

performing initial configuration (Col. 7 Ln. 11 – 15);

creating controls and Web pages (Col. 14 Ln. 30 – 46);

enabling security in Web server applications (Col. 6 Ln. 7 – 9, Ln. 48 – 67, Col. 7 Ln. 50 – 56);

accessing session state values (Col. 6 Ln. 23 – 27);

B. a second group of services related to constructing client applications, the second group of services comprising:

creating windowing graphical user interface environments/enabling graphical functionality (Col. 14 Ln. 30 Ln. 30 – 46);

C. a third group of services related to data and handling XML documents, the third group of services comprising: enabling management of data from multiple data sources (Col. 5 Ln. 25 – 43);

second functions that enable XML processing (Col. 5 Ln. 25 – 37, Col. 10 Ln. 1 – 9, Ln. 59 – 67);

D. a fourth group of services related to base class libraries, the fourth group of services comprising:

defining various collections of objects (Col. 8 Ln. 50 – 67);

inputting and outputting of data (Col. 8 Ln. 29 – 38, Ln. 64 – 67);

enabling a programming interface to network protocols (figure 1 Col. 4 Ln. 60 – 67, Col. 5 Ln. 25 – 37);

enabling system security and permissions (Col. 6 Ln. 7 – 9);

installing and running services (Col. 9 Ln. 27 – 35);

enabling character encoding (inherent in XML language, since XML language supports character encoding); and facilitating runtime operations (Col. 8 Ln. 22 – 28).

Firth teaches customizing data according to cultural related information (Col. 13 Ln 6 – 29);

accessing configuration settings and handling errors in configuration files/debugging and tracing code execution (Col. 13 Ln 6 – 29);

viewing loaded types, methods, and fields; creating, storing and managing various culture-specific resources (Col. 13 Ln. 6 – 29) and

enabling multi-threaded programming (Col. 12 Ln. 52 – 62).

28. Claim 41 is rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 6,792,605 B1 issued to Roberts et al. in view of U.S. Pat. No. 5,987,517 issued to Firth et al. and further in view of U.S. Pat. No. 6,609,158 B1 issued to Nevarez et al.

29. As to claim 41, Robert teaches method implemented at least in part by a computer, for exposing resources using an application program interface, the method comprising:

exposing a first group of services related to creating Web applications, the first group of services comprising:

constructing Web services ("Creating a Web service…" Col. 7 Ln. 50 – 67, "…creating and managing web services…" Col. 9 Ln. 27 – 35);

temporary caching resources ("…caching…" Col. 11 Ln. 1 – 5);

performing initial configuration ("…configuration purposes…" Col. 7 Ln. 11 – 15);;

creating controls and Web pages ("…Pages…" Col. 14 Ln. 30 – 46);

enabling security in Web server applications ("…access privileges…" Col. 6 Ln. 7 – 9, Ln. 48 – 67, Col. 7 Ln. 50 – 56); and

accessing session state values ("…generates a session…" Col. 6 Ln. 23 – 27);

exposing a second group of services related to constructing client applications, the second group of services comprising:

creating windowing graphical user interface environments; and

enabling graphical functionality ("…pages serves as both content and UI that is suitable formatted for display in a browser…" Col. 14 Ln. 30 – 46);

exposing a third group of services related to data and handling XML documents, the third group comprising:

enabling management of data from multiple data sources ("…response data format is XML…" Col. 5 Ln. 25 – 33, Col. 10 Ln. 1 – 9); and

second functions that enable XML processing "…XML response…" Col. 10 Ln. 1 – 9, Ln. 58 – 67);

exposing a fourth group of services related to base class libraries, the fourth group of services comprising:

defining various collections of objects (Directory 102 Col. 4 Ln. 66 – 67, Col. 8 Ln. 50 – 67, Col. 9 Ln. 1 – 25);

inputting and outputting of data ("…Web service Input and output interfaces…" Col. 8 Ln. 29 – 38, Ln. 64 – 67);

enabling a programming interface to network protocols figure 1 Col. 4 Ln. 60 – 67, Col. 5 Ln. 25 – 37);

enabling system security and permissions ("…access privileges…" Col. 6 Ln. 7 – 9, Ln. 48 – 67, Col. 7 Ln. 50 – 56);

installing and running services ("…web services…functionality…" Col. 9 Ln. 27 – 35);

enabling character encoding (inherent in XML language, since XML language supports character encoding); and

facilitating runtime operations ("…runtime…" Col. 8 Ln. 22 – 28); and

Robert is silent with reference to accessing configuration settings and handling errors in configuration files/debugging and tracing code execution (Col. 13 Ln 6 – 29); customizing data according to cultural related information; enabling multi-threaded programming; viewing loaded types, methods, and fields; creating, storing and managing various culture-specific resources; providing a common language runtime layer that translates Web applications written in different programming languages into an intermediate language, the intermediate language being: executed natively by the common language runtime layer; and configured to access resources requested by the client applications; and wherein the different program languages are selected from a plurality of programming languages the plurality of programming languages comprising: Visual Basic; C++; C#; COBOL; Jscript; Perl; Eiffel; and Python.

Firth teaches accessing configuration settings and handling errors in configuration files; debugging and tracing code execution; customizing data according to cultural related information (Col. 13 Ln 6 – 29); enabling multi-threaded programming; viewing loaded types, methods, and fields (Col. 13 Ln. 6 – 29); creating, storing and managing various culture-specific resources (Col. 12 Ln. 52 – 62).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify of system of Roberts with the teaching of Firth because the teaching of Firth would improve the system of Roberts by creating computer network applications by using a library of reentrant network functions which allow an application to reduce the source code required to interact with a computer network such as the internet (Firth Col. 1 Ln. 9 – 14).

Nevarez teaches providing a common language runtime layer that translates Web applications written in different programming languages into an intermediate language, the intermediate language being: executed natively by the common language runtime layer ("...core..." Col. 5 Ln. 5 – 15, UCS Product 224/Core 228 Col. 10 Ln. 5 – 22, "...maps programming languages to a general object interface..." Col. 13 Ln. 58 – 67, "...common object wrapper..." Col. 14 Ln. 7 – 10); and configured to access resources requested by the client applications (Step 310 Col. 12 Ln. 66 – 67, Col. 13 Ln. 5 – 25, "...method invocation..." Col. 13 Ln. 62 – 64, "...allows a user to access..." Col. 14 Ln. 62 – 64); and wherein the different program languages are selected from a plurality of programming languages the plurality of programming languages comprising: Visual Basic; C++; C#; COBOL; Jscript; Perl; Eiffel; and Python (Perl 204/C/C++ 206/Java Script 210/Visual Basic/COBOL Col. 9 Ln. 32 – 40).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify of system of Firth and Roberts with the teaching of Nevarez because the teaching of Nevarez would improve the system of Firth and Roberts by providing a language-vendor independent bridge between components written in different programming languages and allows the components to gain access to specific object models (Nevarez Col. 13 Ln. 58 – 67).

Response to Arguments

Applicant's arguments filed 11/10/08 have been fully considered but they are not persuasive.

Applicant argues in substance that (1) the Chiang prior art does not teach or suggest a common language runtime layer that translates the plurality of applications written in different languages into a common language because the multiple markup languages referenced therein are not programming languages, (2) the Chiang prior art does not teach or suggest the intermediate language being executed natively by a common runtime layer, and (3) the Chiang prior art does not teach or suggest a multi-language application development

The Examiner respectfully traverses Applicant's arguments:

As to point (1), firstly, it may be orderly to define what markup languages are. Markup languages are **programming languages** designed for the processing, definition and presentation of text. The languages specifies code for formatting, both the layout and style, within a text file. The codes used to specify the formatting are called tags. XML and HTML are examples of widely known and used markup languages.

The Chiang prior art discloses a process for web application development. Web applications are generally prepared by graphic designers/business analysts or web developers. The graphic designers/business analysts create the web application screens (graphical user interface) in one of **several available formats**, such as XML and XSL (Extensible Style Language), HTML, cHTML (compact Hypertext Markup Language) and WML **(i.e. different languages)**. These web application screens are then used as input files into a web application server/web application generator. The web application server determines if an application framework code/application source code (intermediate language) is available for the web application, and retrieves the application framework code/application source code from an application directory. If the application framework code/application source code is not available for the web application, then the web application server via the web application generator generates or translates the input files into the application framework code/application source code, along with a business logic foundation code, an event handler skeleton and a graphical user interface code. The web application generator translates the input files (i.e. applications written in different languages) into or generates **same application framework code/application source code regardless of the language format for the input files,** as such making the application framework code/application source code the intermediate language/code and the web application generator the common language layer (because the application framework code/application source code is the same for the all input files). The web application server via the web application generator/event handler executes/process the application framework code/application source code (i.e. application object) in the native language of the application framework code/application source code which could be either **JavaScript, C#, C++ or SmallTalk.**

As to point (2), as indicate above the application framework code/application source code in the intermediate language (i.e. JavaScript, C#, C++ or SmallTalk) because the web application generator generated or translated the input files (i.e. applications written in different languages) into a language (i.e. JavaScript, C#, C++ or SmallTalk) different from the language of the input files and the same for all input files.

As to point (3), again XML, HTML, XSL and WML are different programming languages translated by the web application generator into a common or intermediate language (i.e. JavaScript, C#, C++ or SmallTalk).

Conclusion

Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to CHARLES E. ANYA whose telephone number is (571)272-3757. The examiner can normally be reached on 8:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Meng-Ai An/
Supervisory Patent Examiner, Art Unit 2195

cea.